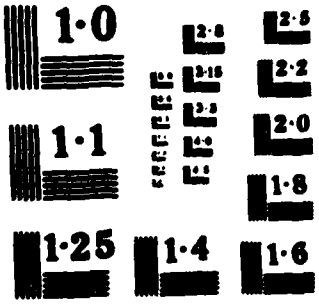
AD-A148 055 MULTIDIMENSIONAL DIGITAL SIGNAL PROCESSING(U) IMPERIAL 1/1
COLL OF SCIENCE AND TECHNOLOGY LONDON (ENGLAND)
T A LANFEAR ET AL. AUG 84 DAJA45-84-C-0025

UNCLASSIFIED F/G 9/2 NL

END


1·0
2·8
2·5
2·2
2·0
1·8
1·1
1·25
1·4
1·6

(13)

MULTIDIMENSIONAL DIGITAL SIGNAL PROCESSING

Final Technical Report

by

T. A. Lanfear

A. G. Constantinides

August 1984

United States Army

EUROPEAN RESEARCH OFFICE OF THE U.S. ARMY

London, England.

CONTRACT NUMBER DAJA45-84-C-0025

Imperial College of Science and Technology- A. G. Constantinides

CONTENTS

1 Introduction

This project has been concerned with the development of software tools for the study, at the most general level, of the ALPS system architecture. The primary aim of these tools is to answer such questions as: is a signal processing application feasible with a given hardware configuration?; will the system degrade gracefully if some of the resources fail?; what is the effect on system performance of changes to details such as the number of resources available, the execution time of a resource etc., etc?

2 ALPS System Simulation

A program has been written in FORTRAN 77 which simulates the ALPS Signal Flow Architecture. It takes the following information into consideration:

1. The signal processing application expressed as a signal flow graph with the modifications to this notation which have been suggested at Imperial College.
2. The number and types of resources in the system.
3. The execution time of the resources.
4. The type of resource associated with each graph node.
5. The number of data busses and their data transfer rates. and
6. The rate at which data is presented to the input of the system.

The simulator then steps through time examining the state of the system and modifying it appropriately so as to simulate the allocation of resources to signal flow graph nodes and the transmission of data along busses. When the state of the system changes a summary of the new system state is printed. The

information presented is:

1 The time.

2 The state of the data busses.

3 The number of resources unused.

4 The bus utilisation as a percentage of the total time.

5 The status of the various resources which are currently in use.

A detailed description of the program and a users manual have been given in the First and Second Interim Reports of this project. A listing of the program is appended to this report together with a floppy written in UNIX tar format containing the program and the following example.

3 An Example

Figure 1 shows a typical signal processing flow graph with two inputs (nodes 1 and 4) and a single output (node 14). Node 0 is the added input node. The prioritised inputs are marked with solid lines and the other inputs with dotted lines.

The table below gives the resource descriptions of the graph.

resource type	execution time	graph node	number
A	250	1,6,7,9	2
B	50	3,10	2
C	12	8	1
D	100	5	1
E	100	14	1
F	25	13	1
G	10	12	1
H	50	11	2
I	1	4	1
J	10	2	1
memory	0	N/A	5

The other pieces of information required are:

bus transfer time = 20

number of busses = 1

From this information the input file can be generated. It is:

```
 14  11   1   20
  0  2  0  0  2  0  0  0  0  0  0  0  0  0  0
  0  0  2  1  0  0  0  0  0  0  2  0  0  0  0
  0  0  0  2  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0
  0  0  0  0  0  2  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  2  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  2  0  0  0
  0  0  0  0  0  0  0  0  2  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  2  0  0  0  0  2
  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  1  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  2  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  2  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  1
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
250 50 12100100 25 10 50  1 10  0
  1 10  2  9  4  1  1  3  1  2  8  7  6  5
```

The output obtained from this example is shown on the following pages.

```
NAME OF FILE TO READ INPUT DATA
GIVE I/P RATE
GIVE TIME FOR FIRST INTERRUPT

TIME IS 0
BUS 1 IS FREE

RESOURCES UNUSED
  1  1  2  1  1  1  1  1  2  0  1  5

BUS UTILISATION
****%

RES NODE   TAG    END    MEM NIP SINK  NODES
  1   1     0     250   0   1   2   3 10
  2   4     0       1   0   1   5


TIME IS 1
BUS 1 IS BUSY

RESOURCES UNUSED
  1  1  2  1  0  1  1  1  2  1  1  5

BUS UTILISATION
 100%

RES NODE   TAG    END    MEM NIP SINK  NODES
  1   1     0     250   0   1   2   3 10
  2   5     0     121   0   1   6


TIME IS 121
BUS 1 IS BUSY

RESOURCES UNUSED
  1  0  2  1  1  1  1  1  2  1  1  5

BUS UTILISATION
  17%

RES NODE   TAG    END    MEM NIP SINK  NODES
  1   1     0     250   0   1   2   3 10
  2   6     0     391   0   1  11


TIME IS 250
BUS 1 IS BUSY

RESOURCES UNUSED
  1  1  1  1  1  1  1  1  2  1  0  4

BUS UTILISATION
  16%

RES NODE   TAG    END    MEM NIP SINK  NODES
  2   6     0     391   0   1  11
  3   2     0     280   0   1   3
  4  10     0  999999   0   1  11
  5   1     0     270   1   1   3
```

TIME IS 26
BUS 1 IS BUSY

RESOURCES UNUSED
1 1 0 1 1 1 1 1 2 1 1 4

BUS UTILISATION
21%

RES	NODE	TAG	END	MEM	NIP	SINK	NODES
2	6	0	391	0	1	11	
4	10	0	999999	0	1	11	
5	1	0	270	1	1	3	
1	3	0	999999	0	1	7	

TIME IS 300
BUS 1 IS BUSY

RESOURCES UNUSED
1 1 0 1 1 1 1 1 2 1 1 5

BUS UTILISATION
27%

RES	NODE	TAG	END	MEM	NIP	SINK	NODES
2	6	0	391	0	1	11	
4	10	0	999999	0	1	11	
1	3	0	370	0	2	7	

TIME IS 370
BUS 1 IS BUSY

RESOURCES UNUSED
1 0 1 1 1 1 1 1 2 1 1 5

BUS UTILISATION
27%

RES	NODE	TAG	END	MEM	NIP	SINK	NODES
2	6	0	391	0	1	11	
4	10	0	999999	0	1	11	
3	7	0	640	0	1	8	

TIME IS 391
BUS 1 IS BUSY

RESOURCES UNUSED
1 1 1 1 1 1 1 1 1 1 1 5

BUS UTILISATION
30%

RES	NODE	TAG	END	MEM	NIP	SINK	NODES
4	10	0	999999	0	1	11	
3	7	0	640	0	1	8	
1	11	0	999999	0	1	12	

```
TIME IS 640
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  1  0  1  1  1  1  1  1  1  5

BUS UTILISATION
  22%

RES NODE   TAG    END    MEM NIP SINK NODES
  4  10     0  999999   0   1  11
  1  11     0  999999   0   1  12
  2   8     0     672   0   1   9 14

TIME IS 672
BUS 1 IS BUSY

RESOURCES UNUSED
  1  1  1  1  1  0  1  1  1  1  1  5

BUS UTILISATION
  22%

RES NODE   TAG    END    MEM NIP SINK NODES
  4  10     0  999999   0   1  11
  1  11     0  999999   0   1  12
  3   9     0     942   0   1  10
  5  14     0  999999   0   1

TIME IS 942
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  1  1  1  0  1  1  1  1  1  5

BUS UTILISATION
  19%

RES NODE   TAG    END    MEM NIP SINK NODES
  4  10     0    1012   0   2  11
  1  11     0  999999   0   1  12
  5  14     0  999999   0   1

TIME IS 1012
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  2  1  1  0  1  1  1  1  1  5

BUS UTILISATION
  19%

RES NODE   TAG    END    MEM NIP SINK NODES
  1  11     0    1082   0   2  12
  5  14     0  999999   0   1
```

```
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  2  1  1  0  1  0  2  1  1  5

BUS UTILISATION
  20%

RES NODE   TAG    END    MEM NIP SINK NODES
  5  14     0   999990   0   1
  2  12     0     1112   0   1  13

TIME IS 1112
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  2  1  1  0  0  1  2  1  1  5

BUS UTILISATION
  21%

RES NODE   TAG    END    MEM NIP SINK NODES
  5  14     0   999999   0   1
  1  13     0     1157   0   1  14


TIME IS 1157
BUS 1 IS BUSY

RESOURCES UNUSED
  1  2  2  1  1  0  1  1  2  1  1  5

BUS UTILISATION
  22%

RES NODE   TAG    END    MEM NIP SINK NODES
  5  14     0     1277   0   2


TIME IS 2000
BUS 1 IS FREE

RESOURCES UNUSED
  1  1  2  1  1  1  1  1  2  0  1  5

BUS UTILISATION
  14%

RES NODE   TAG    END    MEM NIP SINK NODES
  1   1     1     2250   0   1   2  3 10
  2   4     1     2001   0   1   5


    TIME IS 2000

    WHAT DO YOU WISH TO DO?
    OPTIONS ARE:
      1 CHANGE INPUT RATE
      2 CHANGE NUMBER OF RESOURCES IN SYSTEM
      3 CHANGE EXECUTION TIME OF A RESOURCE
      4 CHANGE NUMBER OF BUSSES
      8 STOP PROGRAM
      9 CONTINUE SIMULATION
```

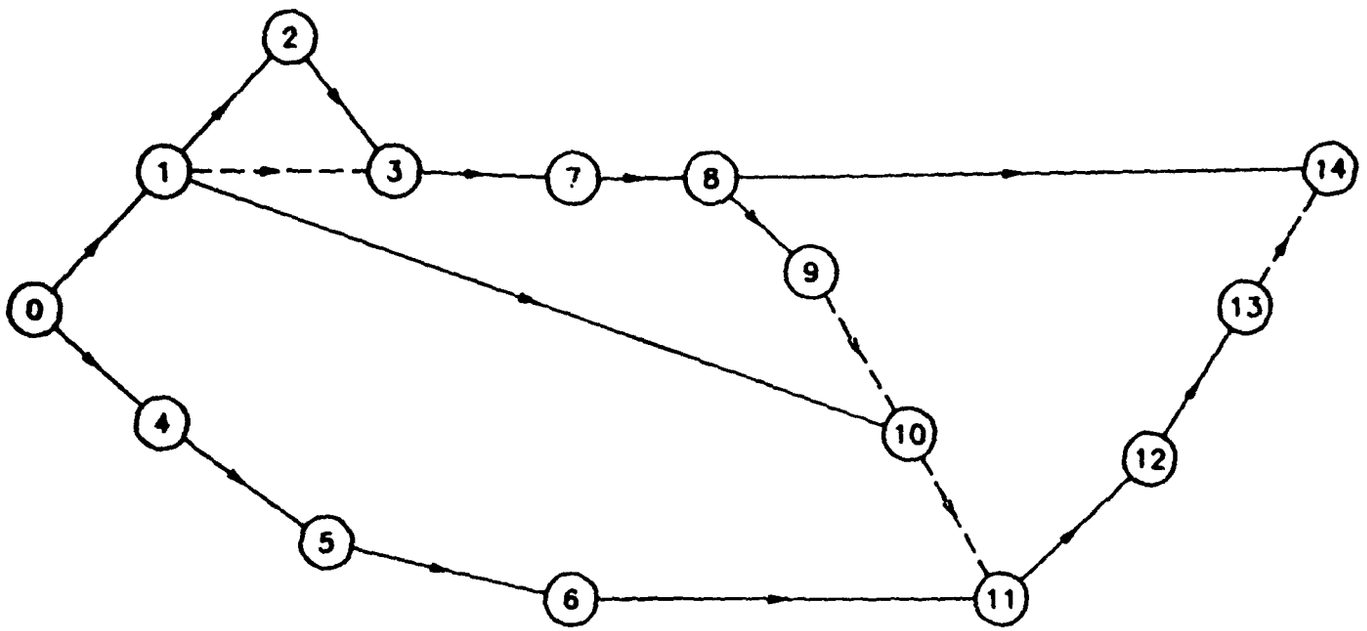


Fig 1 Signal Processing Graph

```
[illegible]
[illegible]

[illegible] STATEMENT FOR [illegible] PROCESSING
[illegible] FOR [illegible]

COMMON [illegible] N1,N2,POOL(N2),EXTIME(N2),TYPE(N2),BFREE(N2),
+FEE(N2,4),[illegible],CONN(N1,N1),NCONN(N1),IF(N1),TRANS(N4,N1+1),
+NTRANS(N4),LIST(N4+1),BUTIL(N2),[illegible]

DO 1 I=1,N2 FOR PRIORITY LINK I (0)
        #1 CRDINAR LINK
        #0 OTHERWISE
POOL(I)=NUMBER OF RESOURCES OF TYPE I
EXTIME(I)=EXECUTION TIME OF RESOURCE TYPE I
TYPE(I)=RESOURCE TYPE OF GRAPH NODE I (INPUT TYPE)
FEE(I,1)=CONFIGURATION OF RESOURCE
   FEE(I,2)=GRAPH NODE BEING EXECUTED ON FEE(I)
   FEE(I,3)=TIME TAG OF DATA BEING EXECUTED (INFINITY IF NO DATA
   FEE(I,4)=TIME WHEN PROCESS WILL FINISH (INFINITY IF FEE IS INACTIVE)
   FEE(I,5)=1 IF RESOURCE IS A MEMORY, 0 OTHERWISE
   FEE(I,6)=NUMBER OF INPUTS WHICH HAVE BEEN SATISFIED
CONN(I,J)=SET OF GRAPH NODES CONNECTED TO I
NCONN(I)=NUMBER OF NODES CONNECTED TO I
BFREE(I)=TIME WHEN POOL I BECOMES FREE
TRANS(I,J)=SET OF GRAPH NODES TO WHICH FEE(I) SHOULD TRANSMIT
NTRANS(I)=NUMBER OF SET TRANS(I,J)
LIST(I)=SET OF RESOURCES WHICH ARE DOING SOMETHING
NIF(I)=NUMBER OF INPUTS TO GRAPH NODE I
NOW=THE CURRENT TIME

MACHINE STANDARD 1 0 UNITS
[illegible]

[illegible]
PARAMETER (N=[illegible])

[illegible]

[illegible] SELECTION ROUTINE
CALL INIT(FEE,NODES,BTYPE,MTYPE,POW,IF,NCONN,NLIST,NIF,NOUT,INRATE,
+          TIME,IW3)

SET TIME FOR FIRST BREAKPOINT
WRITE(NOUT,*) GIVE TIME FOR FIRST INTERRUPT
READ(IN,*) BREAK

THIS IS THE BEGINNING OF THE MAIN PROGRAM LOOP

[illegible] CONTINUE

SET F AS TO CHECK WHETHER SHOULD PRINT
FRINT=0

CHECK WHETHER ANY RESOURCE HAS COMPLETED PROCESSING AN
OUTPUT GRAPH NODE AND SET RESOURCE FREE IF THIS IS SO
DO [illegible] I=1,NLIST
      IF(NCONN(FEE(LIST(I),1)).EQ.0.AND.FEE(LIST(I),3).LE.NOW) THEN
        FEE(LIST(I),1)=0
        FEE(LIST(I),[illegible])=INF
```

```
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]
[illegible]

      REMOVE INACTIVE RESOURCE FROM LIST
      INDEX=0
      DO 40 I=1,NLIST
            IF(LIST(I).NE.INF) THEN
              INDEX=INDEX+1
              LIST(INDEX)=LIST(I)
            ENDIF
   40 CONTINUE
      NLIST=INDEX

C     SET I/P NODE IF IT CONTAINS NO DATA AND TIME IS CORRECT

      IF(MOD(NOW,INTRMT).EQ.0) THEN
        RES(1,1)=0
        RES(1,2)=THG
        RES(1,3)=NOW
        RES(1,4)=1
        RES(1,5)=1
        [illegible]

        SET NODES TO WHICH I/P SHOULD TRANSMIT
        DO 45 I=1,NCOMM
            TRANS(1,I)=COMM(1,I)
   45   CONTINUE
        [illegible]

        ADD I/P RESOURCE TO LIST
        NLIST=NLIST+1
        LIST(NLIST)=1

        INCREMENT TIME TAG
        THG=THG+1

      ENDIF


C     EXAMINE EACH RESOURCE

      DO 20 I=1,NLIST

            IS IT READY TO TRANSMIT
            IF(RES(LIST(I),2).LE.NOW) THEN

              LOCATE A FREE BUS
              IBUS=0
              DO 70 J=1,NBUS
                    IF(BFREE(J).LE.NOW) THEN
                      IBUS=J
                      GO TO 80
                    ENDIF
              CONTINUE

   80         CONTINUE
C             (TRANSMISSIONS FROM RESOURCE 0 ARE ARTIFICIAL
C              AND SO DO NOT NEED BUS)
              IF(IBUS.GT.0.OR.LIST(I).EQ.0) THEN
```

```
SOURCE=GRAPH NODE CORRESPONDING TO RESOURCE
      REQUESTED LIST(I)
SOURCE=REQLIST(I,1)

EXAMINE EACH GRAPH NODE CONNECTED TO SOURCE
TO WHICH TRANSMISSION IS NECESSARY
DO 10 J=1,NTRANS(LIST(I))

      SET NON-COMMON RESOURCE
      IRES=-1
      SINK=GRAPH NODE OF RECEIVER OF INFORMATION
      SINK=TRANS(LIST(I),J)

      TEST=1 IF TRANSMISSION TO SINK IS POSSIBLE
          =0 OTHERWISE
      TEST=1

      CHECK FOR PRIORITY INF
      IF(ADJ(SOURCE,SINK).EQ.1) THEN

        IF PRIORITY INF THEN NEW RESOURCE REQUIRED SO
        CHECK RECEIVE RESOURCE AVAILABLE
        IF(POOL(TYPE(SINK)).EQ.0) TEST=0

      ELSE

        IF NOT PRIORITY INF CHECK FOR ACTIVE RESOURCE
        EXECUTING CORRECT GRAPH NODE AND TIME TAG
        CALL SCHEDTRES(NRES,IRES,TEST,LIST(I),SINK,INRATE)

      ENDIF

      TRANSMIT IF POSSIBLE
      IF(TEST.EQ.1) THEN
        CALL XMIT(NRES,IBUS,NOW,IRES,SINK,LIST(I),INF,TTIME,
                  NLIST
        SET PRINT FLAG
        PRINT=1
      ENDIF

10 CONTINUE

ELIMINATE TRANSFERS THAT HAVE OCCURRED FROM LIST TRANS
INDEX=0
DO 20 I=1,NTRANS(LIST(I))
      IF(TRANS(LIST(I),J).NE.INF) THEN
        INDEX=INDEX+1
        TRANS(LIST(I),INDEX)=TRANS(LIST(I),J)
      ENDIF
CONTINUE
NTRANS(LIST(I))=INDEX

TRANSMIT TO GLOBAL MEMORY IF RESOURCE IS NOT ALREADY
A MEMORY AND THERE ARE STILL TRANSFERS TO MAKE

IF(INDEX.GT.0.AND.RES(LIST(I),4).EQ.0) THEN
  CALL XMITM(NRES,IBUS,NOW,IRES,SOURCE,LIST(I),
  MTYPE,NOUT,TTIME,NLIST,TEST)
  SET PRINT FLAG
  PRINT=1
ENDIF

SET RESOURCE AS FREE AND INCREMENT RESOURCE POOL
IF RESOURCE WAS NOT A MEMORY OR IF THERE ARE NO
```

```
[illegible]
[illegible]
[illegible]
```

```
      [illegible]
      [illegible]
      [illegible]
      [illegible]
      [illegible]

      [illegible] RESOURCE [illegible] HAS BEEN DETERMINED
      IF SRES(IRES,5) EQ [illegible] THEN
        POOL(TYPE(SIN)) = POOL(TYPE(SIN)) +
        [illegible] NODES FOR [illegible]
        DO I = 1, NCONN(SIN)
              TRANS(IRES,I)=CONN(SIN,I)
        CONTINUE
        NTRANS(IRES)=NCONN(SIN)
      END IF

      FREE THE CONDITION FOR STARTING OFF F B NODES
      IF (CONN(SIN,SIN).EQ.1.AND.RES(IRES,5).EQ.MIN(SIN)+1.AND.RES(I,2 .EQ.0)
     -   RES(IRES,5)=RES(IRES,5)+1

      IF (RES(IRES,5).EQ.MIN(SIN)) THEN
        GET FINISH TIME OF PROCESS
        IF(RES(IRES,[illegible]).NE.0) THEN
          [illegible]
          RES(IRES,[illegible])=NOW+EXTIME(RES(SIN,[illegible]))
        ELSE
          [illegible] FREE NODE
          RES(IRES,[illegible])=NOW+EXTIME(RES(SIN,[illegible]))
        END IF
      [illegible]

C     ELIMINATE SIN GRAPH NODE FROM SOURCE RESOURCE TRANSMISSION LIST
      DO 40 I=1,NTRANS(IRES)
          IF(TRANS(IRES,I).EQ.SIN) TRANS(IRES,I)=INF
40    CONTINUE

      [illegible]
      [illegible]


      [illegible] SUBROUTINE MINMARES IRES,CONN,TRANS SOURCE,RES,NTRES,NOUT TRANST,NLIST,
     -                  'EST
      [illegible]

      PARAMETER STATEMENT FOR ARRAY DIMENSIONS
      PARAMETER (N1=20,N2=10,N3=5,N4=50)

      COMMON /A/ NOUT,Q(N1),N1,POOL(N2),ETIME(N2),TYPE(0:N1),BRES(N3),
     +RES(0:N4,5),CONN(0:N1,N1),NCONN(0:N1),NIF(N1),TRANS(0:N4,N1+1),
     +NTRANS(0:N4),LIST(N4+1),BUTIL(N3,2

C     ARE THERE ANY MEMORY UNITS LEFT?
      IF POOL(MTYPE).EQ.0 THEN
        WRITE(NOUT,*) 'OUT OF MEMORY - SYSTEM CRASH
        STOP
      END IF

      IF SOURCE IS I F TYPE THEN BUS IS NOT USED
C     OTHERWISE SET BUS FREE TIME
      IF(SRES.NE.0) SRES(IBUS)=NOW+TRANST

C     LOCATE AN UNUSED ELEMENT IN RESOURCE ARRAY
      DO 10 I=1,NRES
```

```
[illegible]
[illegible]
[illegible]
```

```
[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

SUBROUTINE STATUS(NOUT,NIN,NFREE,NT,NP,IHR,IMIN,ISEC,IFLW
IMPLICIT INTEGER (A-W)
SAVE TEST

PARAMETER STATEMENT FOR AREA DIMENSIONS
PARAMETER (NI=[illegible],NT=5,N4=5)

COMMON /A/ [illegible] POOL(NI),STIME(NI),TYPE(NI),FREE(NI),
+FREE(NI),N4,F1,CONTIN,N1,NOUT,NIN,NI,IFOUT,NIN,NIE(NI),IE(NI,N4),N1=1,
+ [illegible] ,DIST(NI),BUFL(NI),

[illegible] FROM FREE FIRST [illegible]
[illegible] PAUSE
[illegible]

WRITE(NOUT,*)
WRITE(NOUT,*) TIME IS [illegible]
DO 10 I=1,NEW
IF(FREE(I).GE.NOW) WRITE(NOUT,*) BUS [illegible] IS BUSY
IF(FREE(I).LE.NOW) WRITE(NOUT,*) BUS [illegible] IS FREE
[illegible]

WRITE(NOUT,*)
WRITE(NOUT,*) RESOURCES USED
WRITE(NOUT,110) (POOL(I),I=0,RTYPE)

WRITE(NOUT,*)
WRITE(NOUT,*) BUS UTILIZATION
WRITE(NOUT,10) [illegible] (FW-BUTIL(I)) [illegible] I=1,NEW)

WRITE(NOUT,*)
WRITE(NOUT,*) REFERENCE TAB END WRITE [illegible] MODE
DO 11 I=1,NEW
WRITE(NOUT,1[illegible]) [illegible] LIST [illegible] REF [illegible] I [illegible]
[illegible] TRANSLIST [illegible] I=1, [illegible] LIST [illegible]
[illegible]

RETURN

10 FORMAT([illegible],I[illegible],[illegible],I[illegible],[illegible],I4,[illegible],I[illegible],I,[illegible],I1,I[illegible],3I[illegible])
11 FORMAT(1X,2(I[illegible]))
110 FORMAT(1X,3(I4,1H[illegible]))

END


SUBROUTINE MTRNODE(
IMPLICIT INTEGER(A-W)

C PARAMETER STATEMENT FOR AREA DIMENSIONS
PARAMETER (NI=[illegible],NT=5,N4=5)

COMMON /A/ [illegible](NI),POOL(NI),STIME(NI),TYPE(NI),FREE(NI),
+FREE(NI),N4,F1,CONTIN,N1,NOUT,NIN,NI,IFOUT,NIN,NIE(NI),IE(NI,N4),N1=1,
```

```
      SUBROUTINE INITIAL(MODES,RTYPE,MTYPE,NOW,INF,NBUS,NLIST,NIN,NOUT,
     1                   NRATE,TTIME,TAB)
      IMPLICIT INTEGER(H-N)

      ### ENTER ALLOCATION FOR ARRAY DIMENSION
      PARAMETER (N1=20,N2=15,N3=5,N4=50)

      INTEGER A(N1),B(N1),POOL(N2),ETIME(N2),TYPE(N1),WRREE(N3),
     1 [illegible] A(N1,N1),NCOMM(N1),NI(N1),TAHE(N4,N2+1),
     2 [illegible] LIST(N4+1),PUTI(N1,2)

      [illegible]
      [illegible] MODES,RTYPE,NBUS,NIN,NOUT,NRATE,TTIME

      [illegible] OF [illegible] ARRAY MODE
      [illegible]

      [illegible]
      [illegible]

      [illegible]
      MTYPE=0 [illegible]

      NO RESOURCES IN LIST YET
      NLIST=0

      [illegible] IS SPECIAL TYPE
      [illegible]
      [illegible]

      [illegible]
      [illegible]
      [illegible] I=1,NBUS
      [illegible] I=NOW
      [illegible]
      [illegible]
   [illegible] CONTINUE

      FIND TOTAL NUMBER OF RESOURCES

      NRES=0
      DO 5 I=1,RTYPE
           NRES=NRES+POOL(I)
    5 CONTINUE

      INITIALIZE ALL RESOURCES AS DOING NOTHING
      DO 15 I=0,NRES
           RES(I,1)=0
           RES(I,2)=INF
           RES(I,3)=INF
           RES(I,4)=0
           RES(I,5)=0
   15 CONTINUE
```

```
[illegible]
  [illegible]
  [illegible]

[illegible] .EQ.[illegible] THEN
  WRITE(NOUT,*)  [illegible] RESOURCE TYPES- INCREASE PARAMETER IN
  [illegible]

[illegible] .EQ.[illegible] THEN
  WRITE(NOUT,*)  [illegible] BUSSES- INCREASE PARAMETER IN
  [illegible]

[illegible] .EQ.[illegible] THEN
  WRITE(NOUT,*)  [illegible] PROCESSORS- INCREASE PARAMETER IN
  [illegible]

[illegible]
[illegible]


[illegible] (CHANGE(NIN,NOUT,NTUS,INRATE,[illegible],[illegible])
[illegible] [illegible]

PARAMETER STATEMENT FOR ARRAY DIMENSIONS
PARAMETER [illegible] =20,NL=15,NT=5,N4=5[illegible]

[illegible] [illegible],POOL(N2),[illegible],TYPE(N4),BFREE(N[illegible],
[illegible]  [illegible],QUEUE(N1),[illegible],TRANS [illegible] N1+1,
[illegible] [illegible],LIST(N4+1),SUTIL N2,2

PAUSE

[illegible]

[illegible]
[illegible]          TIME IS [illegible]
[illegible]*)
[illegible]*)        WHAT DO YOU WISH TO DO?
[illegible]*)        OPTIONS ARE:
[illegible]*)           1 CHANGE INPUT RATE
[illegible]*)           2 CHANGE NUMBER OF RESOURCES IN SYSTEM
[illegible]*)           3 CHANGE EXECUTION TIME OF [illegible] RESOURCE
[illegible]*)           4 CHANGE NUMBER OF BUSSES
[illegible]*)           5 STOP PROGRAM
[illegible]*)           6 CONTINUE SIMULATION
[illegible]*)
[illegible]*)        ENTER OPTION NUMBER
READ(NIN,*) OPTION

IF(OPTION.EQ.1) THEN
  WRITE(NOUT,*) 'GIVE NEW I/P RATE'
  READ(NIN,*) INRATE

[illegible] .EQ.2 THEN
  WRITE(NOUT,*) 'WHICH RESOURCE ALLOCATION DO YOU WISH TO CHANGE'
  READ(NIN,*) IRES
  WRITE(NOUT,*) 'SPECIFY CHANGE IN RESOURCE POOL FOR RESOURCE ',IRES
  WRITE(NOUT,*) 'POSITIVE NUMBER TO INCREASE POOL, NEGATIVE TO DECREASE
  READ(NIN,*) CHES
  IF(POOL(IRES)+CHES.LT.[illegible] THEN
```

```
[illegible]
[illegible]
[illegible]
```

NL

DATE